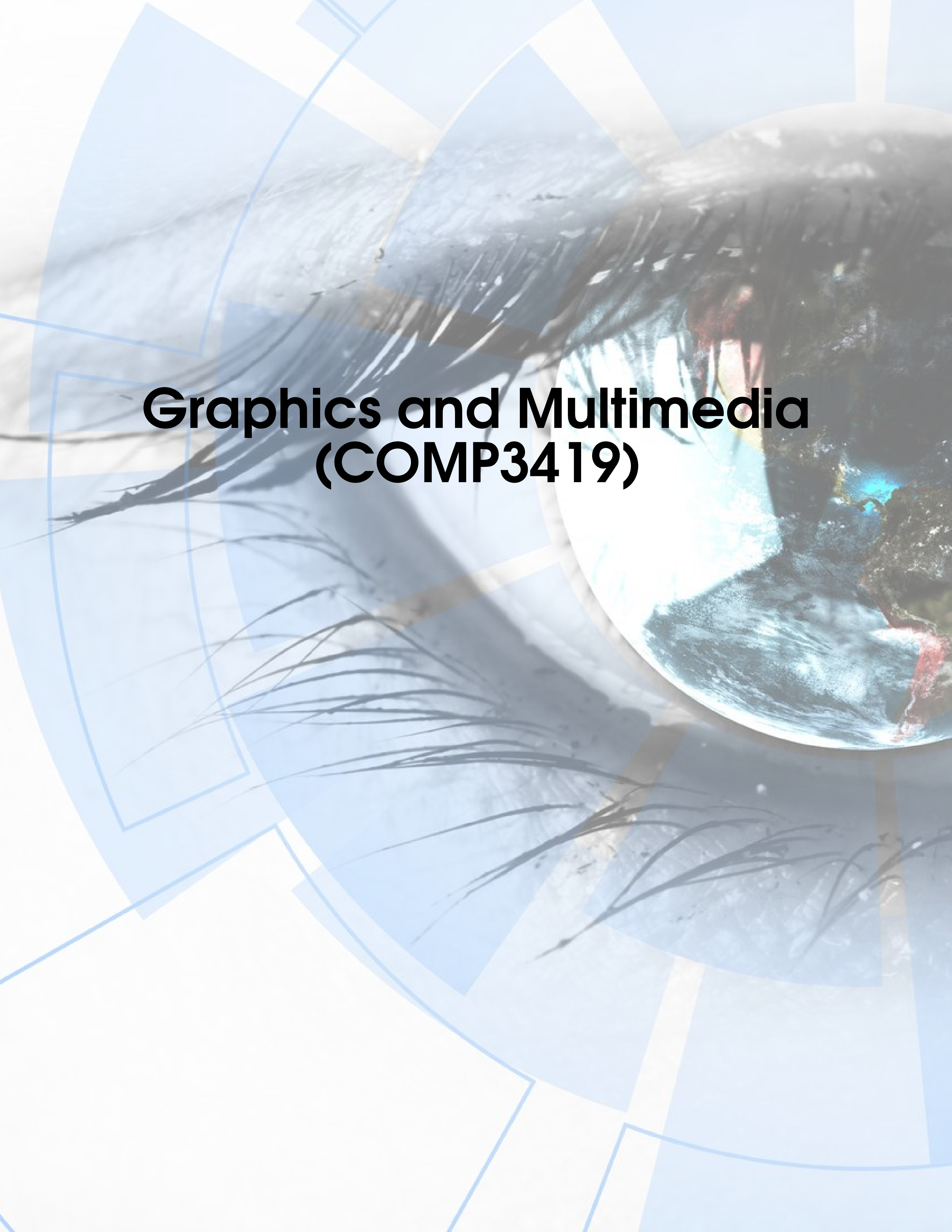


Graphics and Multimedia (COMP3419)

Assignment 1-a Specification

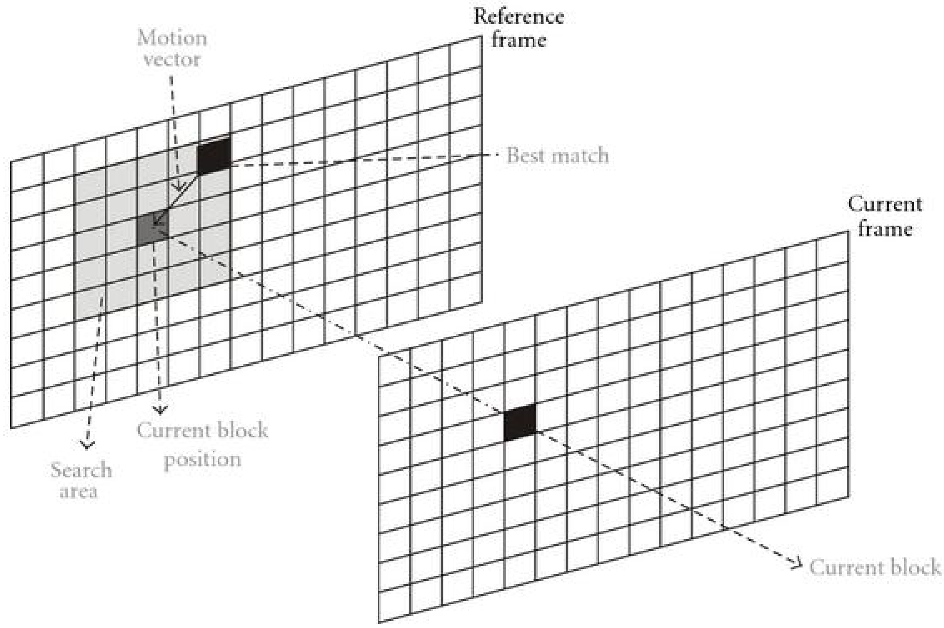
1 Key Information

- The mark of **"COMP3419 Assignment 1-a Motion Estimation and Visualization"** will be given as **two parts**, namely the canvas submission and the live demonstration:
  - **Canvas Submission** [Due Time] before 23:59, Sunday of study vacation week (11 Oct. 2020).
  - **Live Demonstration** [Demo Time] timetable-allocated lab session on Week 7 (12 Oct. 2020).
- This individual assignment is worth **8%** of your final assessment.
- **Submission Deliverables**: Students are asked to create a **zip** file of all deliverables, including all source code. A **README** txt file (to describe the steps/instructions regarding how to get their source code running to derive the expected outputs) should be included within the zip file, to make marker get familiar with the submission.
- Students' assignments **will only be marked if** all deliverables can be **accessed** from the Canvas System, and they can be **runnable** following instructions provided in README txt file. Once plagiarism detected by the Canvas system, the student will receive no mark immediately, as well as other related penalties from university.

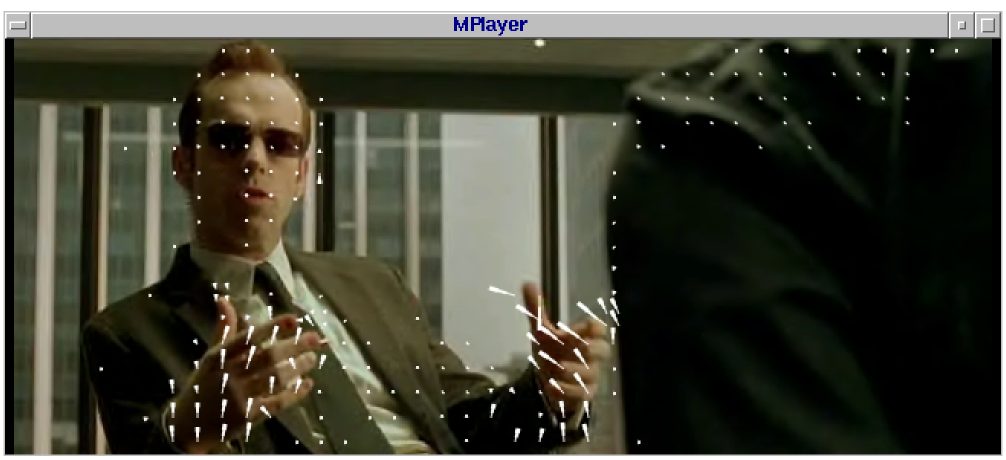
2 Demonstration Rules

- Students are allowed to use any video of framerate $\geq$ **24 fps** to demonstrate their codes. Please have your corresponding output video ready for the demonstration. Video clip within duration **15 - 30 seconds** would be sufficient for demonstration checking.
- Make sure you can demonstrate your program in a reasonable time. Please test the program before coming to the demonstration lab.
- Students will be required to explain the implementation details of their codes to the tutor.
- If students can not finish all the requirements, they should provide a live demo for the workable parts, and seek for partial marks awarded.

3 Motion Estimation and Visualization (Assignment 1-a)



(a) Illustration of the block matching



(b) Example arrow visualization of extracted motion vectors.

Figure 1: The illustration of block matching algorithm and the extracted optical flows.

This assignment is to perform the motion estimation with macroblock matching. The basic premise of motion estimation is that in most cases, consecutive video frames will be similar except for changes induced by objects moving within the frames. The basic idea of motion estimation is to define grids of block regions on two adjacent frames and find the displacement vector (a 2D Cartesian vector in 2D videos) between the matched blocks. To describe the meta-algorithm step by step:

1. Iterate the video frames F_i of size $(f_x,\ f_y)$; Define a grid block of size $(2k+1,\ 2k+1)$, whose size should be odd for convenience in determining the coordinates of centroid point (or, origin point) for each grid block. Each frame F_i results in $\frac{f_x f_y}{(2k+1)^2}$ grid blocks overall.
2. For **each grid block** $B_i^{(x,y)}$ (whose centroid point) at (x,y) in current frame F_i, search for all candidate blocks in next reference frame F_{i+1}, and then find its matched grid block $B_{i+1}^{(x\prime,y\prime)}$ at (x',y') in F_{i+1}.
   - Note: $B_i^{(x,y)}$ is also dubbed as the source block, while $B_{i+1}^{(x\prime,y\prime)}$ is also called as the destination or target block. F_i represents the current frame, while F_{i+1} denotes the next or reference frame. Please see Fig. 1(a) for visual illustration.
   - Among all candidates blocks within the search area of $B_i^{(x,y)}$ in F_{i+1}, the matched grid block $B_{i+1}^{(x\prime,y\prime)}$ should produce the minimum **s**um-of-**s**quared-**d**istances (SSD) to $B_i^{(x,y)}$.
   - Simplifying $B_i^{(x,y)}$ and $B_{i+1}^{(x\prime,y\prime)}$ as B_i and B'_{i+1}, respectively, their **squ**are **r**oot of SSD can be computed as

$$sqr(SSD(B_i, B'_{i+1})) = \sqrt{\sum_{bx=-k}^{k} \sum_{by=-k}^{k} \sum_{bc=0}^{2} \left[B_i(bx,by,bc) - B'_{i+1}(bx,by,bc)\right]^2}, \quad (1)$$

   where bx, by, and bc denote the inside-**b**lock index of pixel location in **x**-direction, **y**-direction, and **c**olor channels (i.e., RGB), respectively. For example, when $bx = by = -k$ and $bc = 0$, then $B_i(bx,by,bc)$ denotes the *R-channel* value of pixel at *top-left corner* of B_i.
3. The (centroid-point) displacement vector from source block (B_i) to target block (B'_{i+1}) can be represented as a 2-D vector, i.e., $(x'-x, y'-y)$. Next, save the displacement vectors for **all source blocks** B_i in frame F_i as a 3D matrix of shape $(\frac{f_x}{2k+1},\ \frac{f_y}{2k+1},\ 2)$.
4. For better visualization purpose, filter out the displacement vectors computed in *Step 3* to remove unexpected noises, whose $sqr(SSD)$ are outside of a self-defined thresholding range $(T_{min},\ T_{max})$.
   - In other words, neglect the noisy displacement vectors whose $sqr(SSD) \notin (T_{min}, T_{max})$.
   - The optimal values of T_{min} and T_{max} vary on different videos or implementation details, which should be determined by experiments.
5. Draw arrows to visualise these selected (centroid-point) displacement fields on frame F_i.
6. Repeat *Step 1-5* for all frames.

R Tip: To speed up for *Step 2*, an assumption could be reached that there are high chances that these matched blocks might appear in positions close to the source block B_i. Therefore, we could search for its neighbouring blocks only within a certain radius R, instead of taking the entire reference frame F_{i+1} as its search area.

R Tip: A helper_function.py is provided to help students draw the arrows in *Step 5*. Please download it from Canvas, and use it as your starting point. Students are also welcome to draw arrows in their preferred shapes, such as the one in Fig. 1b.

Tip: There could be multiple scenes in a video clip, where the inter-scene translations may lead to unpredictable and/or meaningless results produced by our motion estimation algorithm. Hence, students could simply ignore these translation frames and draw nothing on them.

For more details of in-depth understanding of the video estimation algorithm, you may refer to `http://web.stanford.edu/class/ee398a/handouts/lectures/EE398a_MotionEstimation_2012.pdf`

For a faster solution of estimating optical flow with Lucas-Kanade method please see `http://research.ijcaonline.org/volume61/number10/pxc3884611.pdf`

Please note for the assignment submission you are not required to use the Lucas-Kanade method.